Output of Data

This section discusses various aspects of how you can control the format of an output report created with Natural, that is, the way in which the data are displayed.

It covers the following topics:

- Layout of an Output Page - Overview
- Statements DISPLAY and WRITE
- Index Notation (n:n) for Multiple-Value Fields and Periodic Groups
- Page Titles and Page Breaks
- Column Headers
- Parameters to Influence the Output of Fields
- Edit Masks - The EM Parameter
- Vertical Displays

Layout of an Output Page - Overview

The following program illustrates the general layout of an output page:

```
** Example Program 'OUTPUX01'
  DEFINE DATA LOCAL
  1 EMP-VIEW VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 BIRTH
  END-DEFINE
  *
  WRITE TITLE '********** Page Title **********'
  WRITE TRAILER '********** Page Trailer **********'
  AT TOP OF PAGE
    WRITE '===== Top of Page ====='
  END-TOPPAGE
  AT END OF PAGE
    WRITE '===== End of Page ====='
  END-ENDPAGE
  READ (10) EMP-VIEW BY NAME
    DISPLAY NAME FIRST-NAME BIRTH (EM=YYY-MM-DD)
    AT START OF DATA
      WRITE '>>>>> Start of Data >>>>>'
    END-START
    AT END OF DATA
      WRITE '<<<<< End of Data <<<<<'
    END-ENDDATA
  END-READ
  END
```

```
                    ********** Page Title **********
===== Top of Page =====
        NAME              FIRST-NAME         DATE
                                              OF
                                            BIRTH
-------------------- -------------------- ----------

>>>>> Start of Data >>>>>
ABELLAN              KEPA                 1961-04-08
ACHIESON             ROBERT               1963-12-24
ADAM                 SIMONE               1952-01-30
ADKINSON             JEFF                 1951-06-15
ADKINSON             PHYLLIS              1956-09-17
ADKINSON             HAZEL                1954-03-19
ADKINSON             DAVID                1946-10-12
ADKINSON             CHARLIE              1950-03-02
ADKINSON             MARTHA               1970-01-01
ADKINSON             TIMMIE               1970-03-03
<<<<< End of Data <<<<<
                     ********** Page Trailer **********
===== End of Page =====
```

The following statements have an impact on the layout of the report:

Statement	Function
WRITE TITLE	With this statement, you can specify a page title, that is, text to be output at the top of a page.
WRITE TRAILER	With this statement, you can specify a page trailer, that is, text to be output at the bottom of a page.
AT TOP OF PAGE	With this statement, you can specify any processing that is to be performed whenever a new page of the report is started. Any output from this processing will be output below the page title.
AT END OF PAGE	With this statement, you can specify any processing that is to be performed whenever an end-of-page condition occurs. Any output from this processing will be output below any page trailer (as specified with the WRITE TRAILER statement).
AT START OF DATA	With this statement, you specify processing that is to be performed after the first record has been read in a database processing loop. Any output from this processing will be output before the first field value.
AT END OF DATA	With this statement, you specify processing that is to be performed after all records for a processing loop have been processed. Any output from this processing will be output immediately after the last field value.
DISPLAY/WRITE	With these statements, you control the format in which the field values that have been read are to be output.

The statements AT START OF DATA and AT END OF DATA are described in the section Database Access. The other statements listed above are described below.

Statements DISPLAY and WRITE

With the statements DISPLAY and WRITE, you output data and control the format in which information is output.

- DISPLAY Statement
- WRITE Statement
- Column Spacing - The SF Parameter and the nX Notation
- Tab Setting - The nT Notation
- Line Advance - The / Notation

DISPLAY Statement

The DISPLAY statement produces output in column format; that is, the values for one field are output in a column underneath one another. If multiple fields are output, that is, if multiple columns are produced, these columns are output next to one another horizontally.

The order in which fields are displayed is determined by the sequence in which you specify the field names in the DISPLAY statement.

The DISPLAY statement in the following program displays for each person first the personnel number, then the name and then the job title:

```
** Example Program 'DISPLX01'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID NAME JOB-TITLE
  END-READ
  END
```

```
Page      1                                                  99-01-22  11:31:01

  PERSONNEL         NAME                  CURRENT
     ID                                   POSITION
  --------- -------------------- -------------------------

  30020013  GARRET               TYPIST
  30016112  TAILOR               WAREHOUSEMAN
  20017600  PIETSCH              SECRETARY
```

To change the order of the columns that appear in the output report, simply reorder the field names in the DISPLAY statement. For example, if you prefer to list employee names first, then job titles followed by personnel numbers, the appropriate DISPLAY statement would be:

```
** Example Program 'DISPLX02'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    DISPLAY NAME JOB-TITLE PERSONNEL-ID
  END-READ
  END
```



```
Page      1                                                  99-01-22  11:32:06

          NAME                  CURRENT           PERSONNEL
                               POSITION             ID
  -------------------- ------------------------ ---------

  GARRET               TYPIST                   30020013
  TAILOR               WAREHOUSEMAN             30016112
  PIETSCH              SECRETARY                20017600
```

A header is output above each column. Various ways to influence this header are described later in this section.

WRITE Statement

The WRITE statement is used to produce output in free format (that is, not in columns). In contrast to the DISPLAY statement, the following applies to the WRITE statement:

- If necessary, it automatically creates a line advance; that is, a field or text element that does not fit onto the current output line, is automatically output in the next line.
- It does not produce any headers.
- The values of a multiple-value field are output next to one another horizontally, and not underneath one another.

The two example programs on the following page illustrate the basic differences between the DISPLAY statement and the WRITE statement.

You can also use the two statements in combination with one another, as described later in this section.

Example of DISPLAY Statement:

```
** Example Program 'DISPLX03'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 SALARY (1:3)
  END-DEFINE
  READ (2) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME SALARY (1:3)
  END-READ
  END
```

```
Page      1                                                  97-08-14  11:44:00

          NAME              FIRST-NAME        ANNUAL
                                              SALARY
  -------------------- -------------------- ----------

  JONES                VIRGINIA                46000
                                               42300
                                               39300
  JONES                MARSHA                  50000
                                               46000
                                               42700
```

Example of WRITE Statement:


```
** Example Program 'WRITEX01'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 SALARY (1:3)
  END-DEFINE
  READ (2) VIEWEMP BY NAME STARTING FROM 'JONES'
    WRITE NAME FIRST-NAME SALARY (1:3)
  END-READ
  END
```

```
Page      1                                                97-08-14  11:45:00

  JONES                VIRGINIA                 46000      42300
  39300
  JONES                MARSHA                   50000      46000
  42700
```

Column Spacing - The SF Parameter and the nX Notation

By default, the columns output with a DISPLAY statement are separated from one another by *one* space.

With the session parameter SF, you can specify the default number of spaces to be inserted between columns output with a DISPLAY statement. You can set the number of spaces to any value from 1 to 30.

The parameter can be specified with a FORMAT statement to apply to the whole report, or with a DISPLAY statement at statement level, but not at field level.

With the *n***X** notation, you can specify the number of spaces (*n*) to be inserted between two columns.

An *n***X** notation overrides the specification made with the SF parameter.

```
** Example Program 'DISPLX04'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  FORMAT SF=3
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID NAME 5X JOB-TITLE
  END-READ
  END
```

The above example program produces the following output, where the first two columns are separated by 3 spaces due to the SF parameter in the FORMAT statement, while the second and third columns are separated by 5 spaces due to the notation "5X" in the DISPLAY statement:

```
Page      1                                                  99-01-22  11:33:40

  PERSONNEL           NAME                    CURRENT
     ID                                       POSITION
  ---------   --------------------     -------------------------

  30020013    GARRET                   TYPIST
  30016112    TAILOR                   WAREHOUSEMAN
  20017600    PIETSCH                  SECRETARY
```

The *n***X** notation is also available with the WRITE statement to insert spaces between individual output elements:

```
WRITE PERSONNEL-ID 5X NAME 3X JOB-TITLE
```

With the above statement, 5 spaces will be inserted between the fields PERSONNEL-ID and NAME, and 3 spaces between NAME and JOB-TITLE.

Tab Setting - The *n*T Notation

With the *n***T**notation, which is available with the DISPLAY and the WRITE statement, you can specify the position where an output element is to be output.

```
** Example Program 'DISPLX05'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY 5T NAME 30T FIRST-NAME
  END-READ
  END
```

The above program produces the following output, where the field NAME is output starting in the 5th position (counted from the left margin of the page), and the field FIRST-NAME starting in the 30th position:

```
Page      1                                                97-08-21  11:46:01

               NAME                  FIRST-NAME
      --------------------     --------------------
               JONES                 VIRGINIA
               JONES                 MARSHA
               JONES                 ROBERT
```

Line Advance - The / Notation

With a slash "/" in a DISPLAY or WRITE statement, you cause a line advance.

- In a DISPLAY statement, a slash causes a line advance *between fields* and *within text*.
- In a WRITE statement, a slash causes a line advance only when placed *between fields*; within text, it is treated like an ordinary text character.

When placed between fields, the slash must have a blank on either side.

For multiple line advances, you specify multiple slashes.

Example of Line Advance in DISPLAY Statement:

```
** Example Program 'DISPLX06'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 DEPARTMENT
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME / FIRST-NAME 'DEPART-/MENT' DEPARTMENT
  END-READ
  END
```

The above DISPLAY statement produces a line advance after each value of the field NAME and within the text "DEPART-MENT":


```
Page      1                                                  97-08-14  11:45:12

          NAME         DEPART-
       FIRST-NAME       MENT
  -------------------- -------

  JONES                SALE
  VIRGINIA
  JONES                MGMT
  MARSHA
  JONES                TECH
  ROBERT
```

Example of Line Advance in WRITE Statement:

```
** Example Program 'WRITEX02'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 DEPARTMENT
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    WRITE NAME / FIRST-NAME 'DEPART-/MENT' DEPARTMENT //
  END-READ
  END
```

The above WRITE statement produces a line advance after each value of the field NAME, and a double line advance after each value of the field DEPARTMENT, but none within the text "DEPART-/MENT":

```
Page      1                                                  97-08-14  11:45:12

  JONES
  VIRGINIA             DEPART-/MENT SALE


  JONES
  MARSHA               DEPART-/MENT MGMT


  JONES
  ROBERT               DEPART-/MENT TECH
```

Further Examples of DISPLAY and WRITE Statements:

See programs DISPLX13, WRITEX08, DISPLX14, WRITEX09 and DISPLX21 in library SYSEXPG.

Index Notation (*n:n*) for Multiple-Value Fields and Periodic Groups

With the index notation *(n:n)* you can specify how many values of a multiple-value field or how many occurrences of a periodic group are to be output.

For example, the field INCOME in the DDM EMPLOYEES is a periodic group which keeps a record of the annual incomes of an employee for each year he/she has been with the company. These annual incomes are maintained in chronological order. The income of the most recent year is in occurrence "1". If you wanted to have the annual incomes of an employee for the last three years displayed - that is, occurrences "1" to "3" - you would specify the notation "(1:3)" after the field name in a DISPLAY or WRITE statement (as shown in the following example program).

Example of Index Notation in DISPLAY Statement:

```
** Example Program 'DISPLX07'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 INCOME (1:3)
      3 CURR-CODE
      3 SALARY
      3 BONUS (1:1)
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID NAME INCOME (1:3)
    SKIP 1
  END-READ
  END
```

Note that a DISPLAY statement outputs multiple values of a multiple-value field underneath one another:

```
Page      1                                                99-01-22  11:36:58

  PERSONNEL          NAME                       INCOME
     ID
                                  CURRENCY   ANNUAL     BONUS
                                    CODE     SALARY
  --------- -------------------- -------- ---------- ----------

  30020013  GARRET               UKL            4200          0
                                 UKL            4150          0
                                                   0          0

  30016112  TAILOR               UKL            7450          0
                                 UKL            7350          0
                                 UKL            6700          0

  20017600  PIETSCH              USD           22000          0
                                 USD           20200          0
                                 USD           18700          0
```

As a WRITE statement displays multiple values horizontally instead of vertically, this may cause a line overflow and a - possibly undesired - line advance.

If you use only a single field within a periodic group (for example, SALARY) instead of the entire periodic group, and if you also insert a line advance "/" (as shown in the following example between NAME and JOB-TITLE), the report format becomes manageable:

Example of Index Notation in WRITE Statement:

```
** Example Program 'WRITEX03'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
    2 SALARY (1:3)
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    WRITE PERSONNEL-ID NAME / JOB-TITLE SALARY (1:3)
    SKIP 1
  END-READ
  END
```

```
Page      1                                              99-01-22  11:37:18

  30020013 GARRET
  TYPIST                          4200       4150          0

  30016112 TAILOR
  WAREHOUSEMAN                    7450       7350       6700

  20017600 PIETSCH
  SECRETARY                      22000      20200      18700
```


Page Titles and Page Breaks

This section describes various ways of controlling page breaks in a report and the output of page titles at the top of each report page.

- Default Page Title
- Suppress Page Title - The NOTITLE Option
- Define Your Own Page Title - The WRITE TITLE Statement
- Logical Page and Physical Page
- Page Size - The PS Parameter
- Page Advance - The EJ Parameter
- Page Advance - The EJECT and NEWPAGE Statements
- Page Trailer - The WRITE TRAILER Statement
- AT TOP OF PAGE Statement
- AT END OF PAGE Statement

Default Page Title

For each page output via a DISPLAY or WRITE statement, Natural automatically generates a single default title line. This title line contains the page number, the date and the time of day.

```
WRITE 'HELLO'
  END
```

The above program produces the following output with default page title:

```
Page      1                                                97-08-14  18:27:35

  HELLO
```

Suppress Page Title - The NOTITLE Option

If you wish your report to be output without page titles, you add the keyword "NOTITLE" to the DISPLAY or WRITE statement.

```
WRITE NOTITLE 'HELLO'
  END
```

The above program produces the following output without page title:

```
  HELLO
```

Define Your Own Page Title - The WRITE TITLE Statement

If you wish a page title of your own to be output instead of the Natural default page title, you use the statement WRITE TITLE. With this statement, you specify the text for your title (in apostrophes).

```
WRITE TITLE 'THIS IS MY PAGE TITLE'
  WRITE 'HELLO'
  END
```

```
                            THIS IS MY PAGE TITLE
  HELLO
```

With the SKIP option of the WRITE TITLE statement, you can specify the number of empty lines to be output immediately below the title line. After the keyword SKIP, you specify the number of empty lines to be inserted.

```
WRITE TITLE 'THIS IS MY PAGE TITLE' SKIP 2
  WRITE 'HELLO'
  END
```

```
                          THIS IS MY PAGE TITLE


  HELLO
```

SKIP is not only available as part of the WRITE TITLE statement, but also as a stand-alone statement.

By default, the page title is centered on the page and not underlined. However, the WRITE TITLE statement provides the options LEFT JUSTIFIED and UNDERLINED to display the title left-justified and/or underlined.

```
WRITE TITLE LEFT JUSTIFIED UNDERLINED 'THIS IS MY PAGE TITLE' SKIP 2
  WRITE 'HELLO'
  END
```

```
THIS IS MY PAGE TITLE
  ------------------------------------------------------------

  HELLO
```

By default, titles are underlined with a hyphen (-). However, with the UC parameter you can specify another character to be used as underlining character (as described later in this section).

The WRITE TITLE statement is executed whenever a new page is initiated for the report.

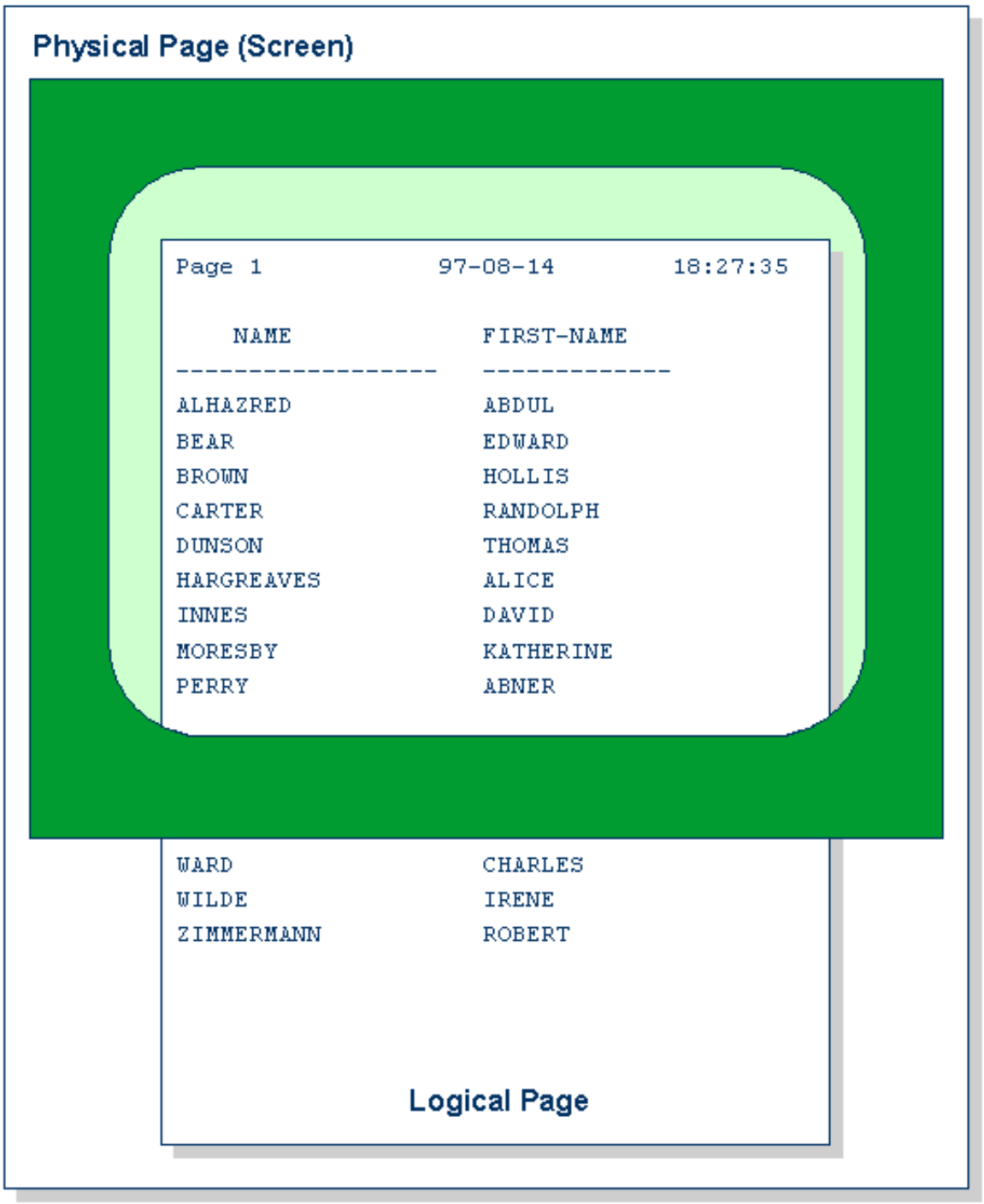
Logical Page and Physical Page

A *logical page* is the output produced by a Natural program.

A *physical page* is your terminal screen on which the output is displayed; or it may be the piece of paper on which the output is printed.

The size of the logical page is determined by the number of lines output by the Natural program.

If more lines are output than fit onto one screen, the logical page will exceed the physical screen, and the remaining lines will be displayed on the next screen.



If information you wish to appear at the bottom of the screen (for example, output created by a WRITE TRAILER or AT END OF PAGE statement) is output on the next screen instead, reduce the logical page size accordingly (with the session parameter PS, which is described below).

Page Size - The PS Parameter

With the parameter PS, you determine the maximum number of lines per (logical) page for a report.

When the number of lines specified with the PS parameter is reached, a page advance occurs (unless page advance is controlled with a NEWPAGE or EJECT statement; see below).

The PS parameter can be set either at session level with the system command GLOBALS, or within a program with the following statements:

- at report level:
  `FORMAT PS=`*`nn`*
- at statement level:
  `DISPLAY (PS=`*`nn`*`)`
  `WRITE (PS=`*`nn`*`)`
  `WRITE TITLE (PS=`*`nn`*`)`
  `WRITE TRAILER (PS=`*`nn`*`)`
  `INPUT (PS=`*`nn`*`)`

Page Advance - The EJ Parameter

With the session parameter EJ, you determine whether page ejects are to be performed or not. By default, EJ=ON applies, which means that page ejects will be performed as specified. If you specify EJ=OFF, page break information will be ignored. This may be useful to save paper during test runs where page ejects are not needed.

The EJ parameter can be set at session level with the system command GLOBALS; for example:

GLOBALS EJ=OFF

Page Advance - The EJECT and NEWPAGE Statements

The EJECT statement overrides the EJ parameter setting. The EJECT statement causes a page advance *without* a title or header line being generated on the next page. A new physical page is started *without* any top-of-page or end-of-page processing being performed (for example, no WRITE TRAILER or AT END OF PAGE , WRITE TITLE, AT TOP OF PAGE or *PAGE-NUMBER processing).

The NEWPAGE statement causes a page advance *with* associated end-of-page and top-of-page processing. A trailer line will be displayed, if specified. A title line, either default or user-specified, will be displayed on the new page (unless the NOTITLE option has been specified in a DISPLAY or WRITE statement).

If the NEWPAGE statement is not used, page advance is automatically controlled by the setting of the PS parameter (see above).

EJECT/NEWPAGE WHEN LESS THAN *n* LINES LEFT

Both the NEWPAGE statement and the EJECT statement provide a WHEN LESS THAN *n*LINES LEFT option. With this option, you specify a number of lines *n*. The NEWPAGE/EJECT statement will then be executed if - at the time the statement is processed - less than *n* lines are available on the current page.

Example:

```
FORMAT PS=55
  ...
  NEWPAGE WHEN LESS THAN 7 LINES LEFT
  ...
```

In this example, the page size is set to 55 lines.

If only 6 or less lines are left on the current page at the time when the NEWPAGE statement is processed, the NEWPAGE statement is executed and a page advance occurs.
If 7 or more lines are left, the NEWPAGE statement is not executed and no page advance occurs; the page advance then occurs depending on the PS parameter, that is, after 55 lines.

NEWPAGE WITH TITLE

The NEWPAGE statement also provides a WITH TITLE option. If this option is not used, a default title will appear at the top of the new page or a WRITE TITLE statement or NOTITLE clause will be executed. The WITH TITLE option of the NEWPAGE statement allows you to override these with a title of your own choice. The syntax of the WITH TITLE option is the same as for the WRITE TITLE statement.

Example:

```
NEWPAGE WITH TITLE LEFT JUSTIFIED 'PEOPLE LIVING IN BOSTON:'
```

The following program illustrates the use of the PS parameter and the NEWPAGE statement. Moreover, the system variable *PAGE-NUMBER is used to display the current page number.

```
** Example Program 'NEWPAX01'
  DEFINE DATA LOCAL
  1 VIEWEMP OF EMPLOYEES
    2 NAME
    2 CITY
    2 DEPT
  END-DEFINE
  FORMAT PS=20
  READ (5) VIEWEMP BY CITY STARTING FROM 'M'
    DISPLAY NAME 'DEPT' DEPT 'LOCATION' CITY
    AT BREAK OF CITY
      NEWPAGE WITH TITLE LEFT JUSTIFIED
        'EMPLOYEES BY CITY - PAGE:' *PAGE-NUMBER
    END-BREAK
  END-READ
  END
```

Note the position of the page breaks and the title line printed on the new page:

```
Page      1                                                  97-08-19  18:27:35

          NAME           DEPT        LOCATION
  -------------------- ------ --------------------

  FICKEN               TECH10 MADISON
  KELLOGG              TECH10 MADISON
  ALEXANDER            SALE20 MADISON
```

```
EMPLOYEES BY CITY - PAGE:      2
          NAME           DEPT        LOCATION
  -------------------- ------ --------------------

  DE JUAN              SALE03 MADRID
  DE LA MADRID         PROD01 MADRID
```


Page Trailer - The WRITE TRAILER Statement

The WRITE TRAILER statement is used to output text (in apostrophes) at the bottom of a page.

```
WRITE TRAILER 'THIS IS THE END OF THE PAGE'
```

The statement is executed when an end-of-page condition is detected, or as a result of a SKIP or NEWPAGE statement.

As the end-of-page condition is checked only *after* an entire DISPLAY or WRITE statement has been processed, it may occur that the logical page size (that is, the number of lines output by a DISPLAY or WRITE statement) causes the physical size of the output page to be exceeded before the WRITE TRAILER statement is executed. To ensure that a page trailer actually appears at the bottom of a physical page, you should set the logical page size (with the PS session parameter) to a value less than the physical page size.

By default, the page trailer is displayed centered on the page and not underlined. However, the WRITE TRAILER statement provides the options LEFT JUSTIFIED and UNDERLINED to display the trailer left-justified and/or underlined:

```
WRITE TRAILER LEFT JUSTIFIED UNDERLINED 'THIS IS THE END OF THE PAGE'
```

AT TOP OF PAGE Statement

The AT TOP OF PAGE statement is used to specify any processing that is to be performed whenever a new page of the report is started.

If the AT TOP OF PAGE processing produces any output, this will be output below the page title (with a skipped line in between). By default, this output is displayed left-justified on the page.

AT END OF PAGE Statement

The AT END OF PAGE statement is used to specify any processing that is to be performed whenever an end-of-page condition occurs.

If the AT END OF PAGE processing produces any output, this will be output after any page trailer (as specified with the WRITE TRAILER statement). By default, this output is displayed left-justified on the page.

The same considerations described above for page trailers regarding physical and logical page sizes and the number of lines output by a DISPLAY or WRITE statement also apply to AT END OF PAGE output.

Further Examples of WRITE TITLE, WRITE TRAILER, AT TOP OF PAGE, AT END OF PAGE and SKIP Statements:

See programs WTITLX01, DISPLX21, ATENPX01, ATTOPX01, SKIPX01 and SKIPX02 in library SYSEXPG.

Further Example of NOTITLE Option:

See program DISPLX20 in library SYSEXPG.

Further Example of NEWPAGE and EJECT Statements:

See program NEWPAX02 in library SYSEXPG.

Column Headers

This section describes various ways of controlling the display of column headers produced by a DISPLAY statement.

- Default Column Headers
- Suppress Default Column Headers - The NOHDR Option
- Define Your Own Column Headers
- Combining NOTITLE and NOHDR
- Centering of Columm Headers - The HC Parameter
- Width of Columm Headers - The HW Parameter
- Filler Characters for Headers - The Parameters FC and GC
- Underlining Character for Titles and Headers - The UC Parameter
- Suppressing Columm Headers - The Notation ’/’

Default Column Headers

By default, each database field output with a DISPLAY statement is displayed with a default column header (which is defined for the field in the DDM).

```
** Example Program 'DISPLX01'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID NAME JOB-TITLE
  END-READ
  END
```

The above example program uses default headers and produces the following output:

```
Page      1                                                    99-01-22  11:31:01

  PERSONNEL         NAME                 CURRENT
     ID                                  POSITION
  --------- -------------------- -------------------------

  30020013  GARRET               TYPIST
  30016112  TAILOR               WAREHOUSEMAN
  20017600  PIETSCH              SECRETARY
```


Suppress Default Column Headers - The NOHDR Option

If you wish your report to be output without column headers, add the keyword "NOHDR" to the DISPLAY statement.

```
DISPLAY NOHDR PERSONNEL-ID NAME JOB-TITLE
```

Define Your Own Column Headers

If you wish column headers of your own to be output instead of the default headers, you specify *'text'* (in apostrophes) immediately before a field,*text* being the header to be used for the field.

```
** Example Program 'DISPLX08'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID
            'EMPLOYEE' NAME
            'POSITION' JOB-TITLE
  END-READ
  END
```

The above program contains the header "EMPLOYEE" for the field NAME, and the header "POSITION" for the field JOB-TITLE; for the field PERSONNEL-ID, the default header is used. The program produces the following output:

```
Page      1                                                  99-01-22  11:39:53

  PERSONNEL       EMPLOYEE               POSITION
     ID
  --------- -------------------- -------------------------

  30020013  GARRET               TYPIST
  30016112  TAILOR               WAREHOUSEMAN
  20017600  PIETSCH              SECRETARY
```

Combining NOTITLE and NOHDR

To create a report that has neither page title nor column headers, you specify the NOTITLE and NOHDR options together in the following order:

```
DISPLAY NOTITLE NOHDR PERSONNEL-ID NAME JOB-TITLE
```

Centering of Column Headers - The HC Parameter

By default, column headers are centered above the columns. With the HC parameter, you can influence the placement of column headers:

- If you specify **HC=L**, headers will be left-justified.
- If you specify **HC=R**, headers will be right-justified.
- If you specify **HC=C**, headers will be centered.

The HC parameter can be used in a FORMAT statement to apply to the whole report, or it can be used in a DISPLAY statement at both statement level and field level.

```
DISPLAY (HC=L) PERSONNEL-ID NAME JOB-TITLE
```

Width of Column Headers - The HW Parameter

With the HW parameter, you determine the width of a column output with a DISPLAY statement.

- If you specify HW=ON, the width of a DISPLAY column is determined by either the length of the header text or the length of the field, whichever is longer. This also applies by default.
- If you specify HW=OFF, the width of a DISPLAY column is determined only by the length of the field. However, HW=OFF only applies to DISPLAY statements which do *not* create headers; that is, either a first DISPLAY statement with NOHDR option or a subsequent DISPLAY statement (see also the Natural Reference documentation).

The HW parameter can be used in a FORMAT statement to apply to the entire report, or it can be used in a DISPLAY statement at both statement level and field level.

Filler Characters for Headers - The Parameters FC and GC

With the FC parameter, you specify the *filler character* which will appear on either side of a *header* produced by a DISPLAY statement across the full column width if the column width is determined by the field length and not by the header (see HW parameter above); otherwise FC will be ignored.

When a group of fields or a periodic group is output via a DISPLAY statement, a *group header* is displayed across all field columns that belong to that group above the headers for the individual fields within the group. With the GC parameter, you can specify the *filler character* which will appear on either side of such a group header.

While the FC parameter applies to the headers of individual fields, the GC parameter applies to the headers for groups of fields.

The parameters FC and GC can be specified in a FORMAT statement to apply to the whole report, or they can be specified in a DISPLAY statement at both statement level and field level.

```
** Example Program 'FORMAX01'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 INCOME (1:1)
      3 CURR-CODE
      3 SALARY
```



```
    3 BONUS (1:1)
END-DEFINE
FORMAT FC=* GC=$
READ (3) VIEWEMP BY NAME
  DISPLAY NAME (FC==) INCOME (1)
END-READ
END
```

The above program produces the following output:

```
Page      1                                                  97-08-19  17:37:27

  ========NAME======== $$$$$$$$$$$$INCOME$$$$$$$$$$$

                       CURRENCY **ANNUAL** **BONUS***
                         CODE     SALARY
  -------------------- -------- ---------- ----------

  ABELLAN              PTA         1450000          0
  ACHIESON             UKL           10500          0
  ADAM                 FRA          159980      23000
```

Underlining Character for Titles and Headers - The UC Parameter

By default, titles and headers are underlined with a hyphen (-).

With the UC parameter, you can specify another character to be used as underlining character.

The UC parameter can be specified in a FORMAT statement to apply to the whole report, or it can be specified in a DISPLAY statement at both statement level and field level.

```
** Example Program 'FORMAX02'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 PERSONNEL-ID
    2 NAME
    2 BIRTH
    2 JOB-TITLE
  END-DEFINE
  FORMAT UC==
  WRITE TITLE LEFT JUSTIFIED UNDERLINED 'EMPLOYEES REPORT' SKIP 1
  READ (3) VIEWEMP BY BIRTH
    DISPLAY PERSONNEL-ID (UC=*) NAME JOB-TITLE
  END-READ
  END
```

In the above program, the UC parameter is specified at program level and at field level: the underlining character specified with the FORMAT statement (=) applies for the whole report - except for the field PERSONNEL-ID, for which a different underlining character (*) is specified. The program produces the following output:

```
EMPLOYEES REPORT
  ============================================================================

  PERSONNEL         NAME                 CURRENT
     ID                                  POSITION
  ********* ==================== =========================

  30020013  GARRET               TYPIST
  30016112  TAILOR               WAREHOUSEMAN
  20017600  PIETSCH              SECRETARY
```


Suppressing Column Headers - The Notation ’/’

With the notation apostrophe-slash-apostrophe (’/’), you can suppress default column headers for individual fields displayed with a DISPLAY statement. While the NOHDR option suppresses the headers of all columns, the notation ’/’ can be used to suppress the header for an individual column.

The notation is specified in the DISPLAY statement immediately before the name of the field for which the column header is to be suppressed.

Compare the following two examples:

```
DISPLAY NAME PERSONNEL-ID JOB-TITLE
```

In this case, the default column headers of all three fields will be displayed:

```
Page      1                                                  97-04-19  17:37:27

          NAME         PERSONNEL          CURRENT
                          ID             POSITION
  -------------------- --------- -------------------------

  ABELLAN              60008339  MAQUINISTA
  ACHIESON             30000231  DATA BASE ADMINISTRATOR
  ADAM                 50005800  CHEF DE SERVICE
  ADKINSON             20008800  PROGRAMMER
  ADKINSON             20009800  DBA
  ADKINSON             20011000  SALES PERSON
```

```
DISPLAY ’/’ NAME PERSONNEL-ID JOB-TITLE
```

In this case, the notation ’/’ causes the column header for the field NAME to be suppressed:

```
Page      1                                                  97-04-19  17:38:45

                       PERSONNEL          CURRENT
                          ID             POSITION
                       --------- -------------------------

  ABELLAN              60008339  MAQUINISTA
  ACHIESON             30000231  DATA BASE ADMINISTRATOR
  ADAM                 50005800  CHEF DE SERVICE
  ADKINSON             20008800  PROGRAMMER
  ADKINSON             20009800  DBA
  ADKINSON             20011000  SALES PERSON
```

Further Examples of Column Headers:

See programs DISPLX15 and DISPLX16 in library SYSEXPG.

Parameters to Influence the Output of Fields

Natural provides several parameters you can use to control the format in which fields are output:

- With the parameters LC, IC and TC, you can specify characters that are to be displayed before or after a field or before a field value.
- With the parameters AL and NL, you can increase or reduce the output length of fields.
- With the parameter SG, you can determine whether negative values are to be displayed with or without a minus sign.
- With the parameter IS, you can suppress the display of subsequent identical field values.
- With the parameter ZP, you can determine whether field values of "0" are to be displayed or not.
- With the parameter ES, you can suppress the display of empty lines generated by a DISPLAY or WRITE statement.

This is discussed in the following topics:

- Leading Characters - The LC Parameter
- Insertion Characters - The IC Parameter
- Trailing Characters - The TC Parameter
- Output Length - The AL and NL Parameters
- Sign Position - The SG Parameter
- Identical Suppress - The IS Parameter
- Zero Printing - The ZP Parameter
- Empty Line Suppression - The ES Parameter

Leading Characters - The LC Parameter

With the LC parameter, you can specify leading characters that are to be displayed immediately *before a field* that is output with a DISPLAY statement. The width of the output column is enlarged accordingly. You can specify 1 to 10 characters.

By default, values are displayed left-justified in alphanumeric fields and right-justified in numeric fields. (These defaults can be changed with the AD parameter; see the Natural Reference documentation). When a leading character is specified for an alphanumeric field, the character is therefore displayed immediately before the field value; for a numeric field, a number of spaces may occur between the leading character and the field value.

The LC parameter can be used with the following statements: FORMAT and DISPLAY. It can be set at statement level and at field level.

Insertion Characters - The IC Parameter

With the IC parameter, you specify the characters to be inserted in the column immediately *preceding the value of a field* that is output with a DISPLAY statement. You can specify 1 to 10 characters.

For a numeric field, the insertion characters will be placed immediately before the first significant digit that is output, with no intervening spaces between the specified character and the field value. For alphanumeric fields, the effect of the IC parameter is the same as that of the LC parameter.

The parameters LC and IC cannot both be applied to one field.

The IC parameter can be used with the following statements: FORMAT and DISPLAY. It can be set at statement level and at field level.

Trailing Characters - The TC Parameter

With the TC parameter, you can specify trailing characters that are to be displayed immediately *to the right of a field* that is output with a DISPLAY statement. The width of the output column is enlarged accordingly. You can specify 1 to 10 characters.

The TC parameter can be used with the following statements: FORMAT and DISPLAY. It can be set at statement level and at field level.

Output Length - The AL and NL Parameters

With the AL parameter, you can specify the *output length* for an alphanumeric field; with the NL parameter, you can specify the *output length* for a numeric field. This determines the length of a field as it will be output, which may be shorter or longer than the actual length of the field (as defined in the DDM for a database field, or in the DEFINE DATA statement for a user-defined variable).

Both parameters can be used with the following statements: FORMAT, DISPLAY, WRITE, and INPUT. They can be set at statement level and at field level.

Note:
If an edit mask is specified, it overrides an NL or AL specification. Edit masks are described later in this section.

Sign Position - The SG Parameter

With the SG parameter, you can determine whether or not a sign position is to be allocated for numeric fields.

- By default, SG=ON applies, which means that a sign position is allocated for numeric fields.
- If you specify SG=OFF, negative values in numeric fields will be output without a minus sign (-).

The SG parameter can be used with the following statements: FORMAT, DISPLAY, WRITE, and INPUT. It can be set at both statement level and field level.

Note:
If an edit mask is specified, it overrides an SG specification. Edit masks are described later in this section.

Example Program without Parameters:

```
** Example Program 'FORMAX03'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 SALARY (1:1)
    2 BONUS  (1:1,1:1)
  END-DEFINE
  READ (5) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME
            SALARY (1:1) BONUS (1:1,1:1)
  END-READ
  END
```

The above program contains no parameter settings and produces the following output:

```
Page      1                                                97-08-15  17:25:19

          NAME              FIRST-NAME        ANNUAL     BONUS
                                              SALARY
  ------------------- ------------------- ---------- ----------

  JONES               VIRGINIA                 46000       9000
  JONES               MARSHA                   50000          0
  JONES               ROBERT                   31000          0
  JONES               LILLY                    24000          0
  JONES               EDWARD                   37600          0
```


Example Program with Parameters AL, NL, LC, IC and TC:

```
** Example Program 'FORMAX04'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 SALARY (1:1)
    2 BONUS  (1:1,1:1)
  END-DEFINE
  FORMAT AL=10 NL=6
  READ (5) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME (LC=*) FIRST-NAME (TC=*)
            SALARY (1:1)(IC=$) BONUS (1:1,1:1)(LC=>)
  END-READ
  END
```

The above program produces the following output. Compare the layout of this output with that of the previous program to see the effect of the individual parameters:

```
Page      1                                                    97-08-19  17:26:12

     NAME      FIRST-NAME   ANNUAL   BONUS
                            SALARY
  ----------- ----------- -------- --------

  *JONES      VIRGINIA  *   $46000 >   9000
  *JONES      MARSHA    *   $50000 >      0
  *JONES      ROBERT    *   $31000 >      0
  *JONES      LILLY     *   $24000 >      0
  *JONES      EDWARD    *   $37600 >      0
```

As you can see in the above example, any output length you specify with the AL or NL parameter does not include any characters specified with the LC, IC and TC parameters: the width of the NAME column, for example, is 11 characters - 10 for the field value (AL=10) plus 1 leading character.

The width of the SALARY and BONUS columns is 8 characters - 6 for the field value (NL=6), plus 1 leading/inserted character, plus 1 sign position (because SG=ON applies).

Identical Suppress - The IS Parameter

With the IS parameter, you can suppress the display of identical information in successive lines created by a WRITE or DISPLAY statement.

- By default, IS=OFF applies, which means that identical field values will be displayed.
- If IS=ON is specified, a value which is identical to the previous value of that field will not be displayed.

The IS parameter can be specified with a FORMAT statement to apply to the whole report, or it can be specified in a DISPLAY or WRITE statement at both statement level and field level.

The effect of the parameter IS=ON can be suspended for one record by using the statement SUSPEND IDENTICAL SUPPRESS; see the Natural Statements documentation for details.

Compare the output of the following two example programs to see the effect of the IS parameter. In the second one, the display of identical values in the NAME field is suppressed.

Example Program without IS Parameter:

```
** Example Program 'FORMAX05'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME
  END-READ
  END
```

```
Page      1                                                  97-08-18  17:25:19

          NAME              FIRST-NAME
  ------------------- --------------------

  JONES               VIRGINIA
  JONES               MARSHA
  JONES               ROBERT
```

Example Program with IS Parameter:

```
** Example Program 'FORMAX06'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
  END-DEFINE
  FORMAT IS=ON
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME
  END-READ
  END
```



```
Page      1                                                  97-08-18  17:26:02

          NAME              FIRST-NAME
  ------------------- --------------------

  JONES               VIRGINIA
                      MARSHA
                      ROBERT
```

Zero Printing - The ZP Parameter

With the ZP parameter, you determine how a field value of zero is to be displayed.

- By default, ZP=ON applies, which means that one "0" (for numeric fields) or all zeros (for time fields) will be displayed for each field value that is zero.
- If you specify ZP=OFF, the display of each field value which is zero will be suppressed.

The ZP parameter can be specified with a FORMAT statement to apply to the whole report, or it can be specified in a DISPLAY or WRITE statement at both statement level and field level.

Empty Line Suppression - The ES Parameter

With the ES parameter, you can suppress the output of empty lines created by a DISPLAY or WRITE statement.

- By default, ES=OFF applies, which means that lines containing all blank values will be displayed.
- If ES=ON is specified, a line resulting from a DISPLAY or WRITE statement which contains all blank values will not be displayed. This is particularly useful when displaying multiple-value fields or fields which are part of a periodic group if a large number of empty lines are likely to be produced.

The ES parameter can be specified with a FORMAT statement to apply to the whole report, or it can be specified in a DISPLAY or WRITE statement at statement level.

Note:
To achieve empty suppression for numeric values, in addition to ES=ON the parameter ZP=OFF must also be set for the fields concerned in order to have null values turned into blanks and thus not output either.

Compare the output of the following two example programs to see the effect of the parameters ZP and ES.

Example Program without Parameters ZP and ES:

```
** Example Program 'FORMAX07'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 BONUS (1:2,1:1)
  END-DEFINE
  READ (4) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME BONUS (1:2,1:1)
  END-READ
  END
```

```
Page      1                                                  97-08-18  17:26:19

          NAME              FIRST-NAME        BONUS
  ------------------- -------------------- ----------

  JONES               VIRGINIA                  9000
                                                6750
  JONES               MARSHA                       0
                                                   0
  JONES               ROBERT                       0
                                                   0
  JONES               LILLY                        0
                                                   0
```

Example Program with Parameters ZP and ES:

```
** Example Program 'FORMAX08'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 BONUS (1:2,1:1)
  END-DEFINE
  FORMAT ES=ON
  READ (4) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY NAME FIRST-NAME BONUS (1:2,1:1)(ZP=OFF)
  END-READ
  END
```

```
Page      1                                                  97-08-18  17:27:12

          NAME              FIRST-NAME        BONUS
  ------------------- -------------------- ----------

  JONES               VIRGINIA                  9000
                                                6750
  JONES               MARSHA
  JONES               ROBERT
  JONES               LILLY
```

Further Examples of Parameters LC, IC, TC, AL, NL, IS, ZP and ES, and SUSPEND IDENTICAL SUPPRESS Statement:

See programs DISPLX17, DISPLX18, DISPLX19 SUSPEX01, SUSPEX02 and COMPRX03 in library SYSEXPG.

Edit Masks - The EM Parameter

With the EM parameter you can specify an *edit mask* for an alphanumeric or numeric field, that is, determine character by character the format in which the field values are to be output.

Example:

```
DISPLAY NAME (EM=X^X^X^X^X^X^X^X^X^X)
```

In this example, each "X" represents one character of an alphanumeric field value to be displayed, and each "^" represents a blank. If displayed via the above DISPLAY statement, the name "JOHNSON" would appear as follows:

```
J O H N S O N
```

You can specify the EM parameter at report level (in a FORMAT statement), at statement level (in a DISPLAY, WRITE, INPUT, MOVE EDITED or PRINT statement) or at field level (in a DISPLAY, WRITE or INPUT statement).

An edit mask specified with the EM parameter will override a default edit mask specified for a field in the DDM. If EM=OFF is specified, no edit mask at all will be used. An edit mask specified at statement level will override an edit mask specified at report level. An edit mask specified at field level will override an edit mask specified at statement level.

The following topics are covered below:

- Edit Masks for Numeric Fields
- Edit Masks for Alphanumeric Fields
- Length of Fields
- Edit Masks for Date and Time Fields
- Examples of Edit Masks

Edit Masks for Numeric Fields

Edit masks for numeric fields (formats N, I, P, F) must include a "9" for each output position you want filled with a number (even if it is zero). A "Z" is used to indicate that the output position will be filled only if the available number is not zero. A decimal point is indicated with a period "." To the right of the decimal point, a "Z" must not be specified. Leading, trailing, and insertion characters - for example, sign indicators - can be added.

Edit Masks for Alphanumeric Fields

Edit masks for alphanumeric fields must include an "X" for each alphanumeric character that is to be output. With a few exceptions, you may add leading, trailing and insertion characters (with or without enclosing them in apostrophes).

The character "^" is used to insert blanks in edit mask for both numeric and alphanumeric fields.

Length of Fields

It is important to be aware of the length of the field to which you assign an edit mask. If the edit mask is longer than the field, this will yield unexpected results. If the edit mask is shorter than the field, the field output will be truncated to just those positions specified in the edit mask.

Examples:

Assuming an alphanumeric field that is 12 characters long and the field value to be output is "JOHNSON", the following edit masks will yield the following results:

```
EM=X.X.X.X.X            Output:    J.O.H.N.S

EM=****XXXXXX****       Output:    ****JOHNSO**
```

Edit Masks for Date and Time Fields

Edit masks for date fields can include the characters "D" (day), "M" (month) and "Y" (year) in various combinations. Edit masks for time fields can include the characters "H" (hour), "I" (minute), "S" (second) and "T" (tenth of a second) in various combinations.

In conjunction with edit masks for date and time fields, see also the date and time system variables.

Examples of Edit Masks

Some examples of edit masks, along with possible output they produce, are provided below. In addition, the abbreviated notation for each edit mask is given. You can use either the abbreviated or the long notation.

Edit Mask	Abbreviation	Output A	Output B
EM=999.99	EM=9(3).9(2)	367.32	005.40
EM=ZZZZZ9	EM=Z(5)9(1)	0	579
EM=X^XXXXX	EM=X(1)^X(5)	B LUE	A 19379
EM=XXX...XX	EM=X(3)...X(2)	BLU...E	AAB...01
EM=MM.DD.YY	*	01.05.87	12.22.86
EM=HH.II.SS.T	**	08.54.12.7	14.32.54.3

* Use a date system variable.

** Use a time system variable.

For further information about edit masks, see the session parameter EM in the Natural Reference documentation.

Example Program without EM Parameters:

```
** Example Program 'EDITMX01'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 JOB-TITLE
    2 SALARY (1:3)
    2 CITY
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY 'N A M E' NAME /
            'OCCUPATION' JOB-TITLE
            'SALARY' SALARY (1:3)
            'LOCATION' CITY
   SKIP 1
  END-READ
  END
```

The above program produces the following output which shows the default edit masks available:

```
Page      1                                                  97-08-19  17:26:19

           N A M E            SALARY         LOCATION
         OCCUPATION
  ------------------------ ---------- --------------------

  JONES                          46000 TULSA
  MANAGER                        42300
                                 39300

  JONES                          50000 MOBILE
  DIRECTOR                       46000
                                 42700

  JONES                          31000 MILWAUKEE
  PROGRAMMER                     29400
                                 27600>
```

Example Program with EM Parameters:

```
** Example Program 'EDITMX02'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 JOB-TITLE
    2 SALARY (1:3)
  END-DEFINE
  READ (3) VIEWEMP BY NAME STARTING FROM 'JONES'
    DISPLAY 'N A M E' NAME  (EM=X^X^X^X^X^X^X^X^X^X^X^X^X^X^X^X) /
            FIRST-NAME      (EM=...X(10)...)
            'OCCUPATION' JOB-TITLE (EM=' ___ 'X(12))
            'SALARY' SALARY (1:3) (EM=' USD 'ZZZ,999)
    SKIP 1
  END-READ
  END
```

The above program produces the following output. Compare the output with that of the previous program to see how the EM specifications affect the way the fields are displayed.

```
Page      1                                                97-08-19  17:26:29

             N A M E                OCCUPATION       SALARY
           FIRST-NAME
  ------------------------------ ----------------- -----------

  J O N E S                      ___ MANAGER       USD  46,000
  ..VIRGINIA  ...                                  USD  42,300
                                                   USD  39,300

  J O N E S                      ___ DIRECTOR      USD  50,000
  ..MARSHA    ...                                  USD  46,000
                                                   USD  42,700

  J O N E S                      ___ PROGRAMMER    USD  31,000
  ..ROBERT    ...                                  USD  29,400
                                                   USD  27,600
```

Further Examples of Edit Masks:

See programs EDITMX03, EDITMX04 and EDITMX05 in library SYSEXPG.

Vertical Displays

There are two ways of creating vertical displays:

- You can use a combination of the statements DISPLAY and WRITE.
- You can use the VERT option of the DISPLAY statement.

The following topics are covered below:

- Combining DISPLAY and WRITE
- The Tab Notation T*-field
- The Positioning Notation x/y
- The DISPLAY VERT Statement
- The Tab Notation P*-field

Combining DISPLAY and WRITE

As described earlier in this section, the DISPLAY statement normally presents the data in columns with default headers, while the WRITE statement presents data horizontally without headers.

You can combine the features of the two statements to produce vertical displays of field values.

The DISPLAY statement produces the values of different fields for the same record across the page with a column for each field. The field values for each record are displayed below the values for the previous record.

By using a WRITE statement after a DISPLAY statement, you can insert textand/or field values specified in the WRITE statement between records displayed via the DISPLAY statement.

The following program illustrates the combination of DISPLAY and WRITE:

```
** Example Program 'WRITEX04'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 JOB-TITLE
    2 CITY
    2 DEPT
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'SAN FRANCISCO'
    DISPLAY NAME JOB-TITLE
    WRITE 20T 'DEPT:' DEPT
    SKIP 1
  END-READ
  END
```


It produces the following output:

```
Page      1                                                    97-08-19  17:52:19

          NAME                  CURRENT
                                POSITION
  -------------------- -------------------------

  KOLENCE                 MANAGER
                        DEPT: TECH05

  GOSDEN                  ANALYST
                        DEPT: TECH10

  WALLACE                 SALES PERSON
                        DEPT: SALE20
```


Tab Notation T*_field_

In the previous example, the position of the field DEPT is determined by the tab notation *n***T** (in this case "20T", which means that the display begins in column 20 on the screen).

Field values specified in a WRITE statement can be lined up automatically with field values specified in the first DISPLAY statement of the program by using the tab notation **T***_field_ (where *field* is the name of the field to which the field is to be aligned).

In the following program, the output produced by the WRITE statement is aligned to the field JOB-TITLE by using the notation "T*JOB-TITLE":

```
** Example Program 'WRITEX05'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 JOB-TITLE
    2 DEPT
    2 CITY
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'SAN FRANCISCO'
    DISPLAY NAME JOB-TITLE
    WRITE T*JOB-TITLE 'DEPT:' DEPT
    SKIP 1
  END-READ
  END
```

```
Page      1                                                97-08-19  17:52:19

          NAME                 CURRENT
                               POSITION
  -------------------- ------------------------

  KOLENCE              MANAGER
                       DEPT: TECH05

  GOSDEN               ANALYST
                       DEPT: TECH10

  WALLACE              SALES PERSON
                       DEPT: SALE20
```

Positioning Notation *x/y*

When you use the DISPLAY and WRITE statements in sequence and multiple lines are to be produced by the WRITE statement, you can use the notation *x/y* (number-slash-number) to determine in which row/column something is to be displayed. The positioning notation causes the next element in the DISPLAY or WRITE statement to be placed **x** lines below the last output, beginning in column *y* of the output.

The following program illustrates the use of this notation:

```
** Example Program 'WRITEX06'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 MIDDLE-I
    2 ADDRESS-LINE (1:1)
    2 CITY
    2 ZIP
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'NEW YORK'
    DISPLAY 'NAME AND ADDRESS' NAME
    WRITE   1/5  FIRST-NAME        1/30 MIDDLE-I
            2/5  ADDRESS-LINE (1:1)
            3/5  CITY              3/30 ZIP /
  END-READ
  END
```

```
Page      1                                                  97-08-19  17:55:47

    NAME AND ADDRESS
  --------------------

  RUBIN
      SYLVIA                    L
      2003 SARAZEN PLACE
      NEW YORK                  10036

  WALLACE
      MARY                      P
      12248 LAUREL GLADE C
      NEW YORK                  10036

  KELLOGG
      HENRIETTA                 S
      1001 JEFF RYAN DR.
      NEWARK                    19711
```


DISPLAY VERT Statement

The standard display mode in Natural is horizontal. With the VERT clause option of the DISPLAY statement, you can override the standard display and produce a vertical field display. The HORIZ clause option, which can be used in the same DISPLAY statement, re-activates the standard horizontal display mode.

Column headings in vertical mode are controlled with various forms of the AS clause:

- Without AS clause, no column headings will be output.
- AS CAPTIONED causes default headings to be displayed.
- AS *text* causes the specified *text* to be displayed as column heading. Note that a slash (/) within the *text* element in a DISPLAY statement causes a line advance.
- AS *text* CAPTIONED causes the specified *text* to be displayed as column heading, and the default column headings to be displayed immediately before the field value in each line that is output.

The following example programs illustrate the use of the DISPLAY VERT statement.

DISPLAY VERT without AS Clause

The following program has no AS clause, which means that no column headings are output.

```
** Example Program 'DISPLX09'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 CITY
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'NEW YORK'
    DISPLAY VERT NAME FIRST-NAME / CITY
    SKIP 2
  END-READ
  END
```

Note that all field values are displayed vertically underneath one another:

```
Page      1                                                  97-08-19  17:55:47

  RUBIN
  SYLVIA

  NEW YORK


  WALLACE
  MARY

  NEW YORK


  KELLOGG
  HENRIETTA

  NEWARK
```

DISPLAY VERT AS CAPTIONED and HORIZ

The following program contains a VERT and a HORIZ clause, which causes some column values to be output vertically and others horizontally; moreover AS CAPTIONED causes the default column headers to be displayed.

```
** Example Program 'DISPLX10'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 CITY
    2 JOB-TITLE
    2 SALARY (1:1)
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'NEW YORK'
    DISPLAY VERT AS CAPTIONED NAME FIRST-NAME
            HORIZ JOB-TITLE SALARY (1:1)
    SKIP 1
  END-READ
  END
```

```
Page      1                                                97-08-19  17:55:47

          NAME                  CURRENT            ANNUAL
       FIRST-NAME              POSITION            SALARY
  -------------------- ------------------------- ----------

  RUBIN                SECRETARY                      17000
  SYLVIA

  WALLACE              ANALYST                        38000
  MARY

  KELLOGG              DIRECTOR                       52000
  HENRIETTA
```


DISPLAY VERT AS *text*

The following program contains an AS *text* clause, which displays the specified *text* as column header.

```
** Example Program 'DISPLX11'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 CITY
    2 JOB-TITLE
    2 SALARY (1:1)
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'NEW YORK'
    DISPLAY VERT AS 'EMPLOYEES' NAME FIRST-NAME
            HORIZ JOB-TITLE SALARY (1:1)
    SKIP 1
  END-READ
  END
```

```
Page      1                                                  97-08-19  7:55:47

       EMPLOYEES                CURRENT            ANNUAL
                               POSITION            SALARY
  -------------------- ------------------------- ----------

  RUBIN                SECRETARY                      17000
  SYLVIA

  WALLACE              ANALYST                        38000
  MARY

  KELLOGG              DIRECTOR                       52000
  HENRIETTA
```

DISPLAY VERT AS *text* CAPTIONED

The following program contains an AS *text* CAPTIONED clause.

```
** Example Program 'DISPLX12'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 NAME
    2 FIRST-NAME
    2 CITY
    2 JOB-TITLE
    2 SALARY (1:1)
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'NEW YORK'
    DISPLAY VERT AS 'EMPLOYEES' CAPTIONED NAME FIRST-NAME
            HORIZ JOB-TITLE SALARY (1:1)
    SKIP 1
  END-READ
  END
```

This clause causes the default column headers (NAME and FIRST-NAME) to be placed before the field values:

```
Page      1                                                97-04-19  17:55:47

             EMPLOYEES                      CURRENT            ANNUAL
                                           POSITION            SALARY
  ------------------------------ ------------------------- ----------

  NAME RUBIN                     SECRETARY                      17000
  FIRST-NAME SYLVIA

  NAME WALLACE                   ANALYST                        38000
  FIRST-NAME MARY

  NAME KELLOGG                   DIRECTOR                       52000
  FIRST-NAME HENRIETTA
```


Tab Notation P**field*

If you use a combination of DISPLAY VERT statement and subsequent WRITE statement, you can use the tab notation **P***field* in the WRITE statement to align the position of a field to the column *and* line position of a particular field specified in the DISPLAY VERT statement.

In the following program, the fields SALARY and BONUS are displayed in the same column, SALARY in every first line, BONUS in every second line.
The text "***SALARY PLUS BONUS***" is aligned to SALARY, which means that it is displayed in the same column as SALARY and in the first line, whereas the text "(IN US DOLLARS)" is aligned to BONUS and therefore displayed in the same column as BONUS and in the second line.

```
** Example Program 'WRITEX07'
  DEFINE DATA LOCAL
  1 VIEWEMP VIEW OF EMPLOYEES
    2 CITY
    2 NAME
    2 JOB-TITLE
    2 SALARY (1:1)
    2 BONUS  (1:1,1:1)
  END-DEFINE
  READ (3) VIEWEMP BY CITY STARTING FROM 'LOS ANGELES'
    DISPLAY NAME JOB-TITLE VERT AS 'INCOME' SALARY (1) BONUS (1,1)
    WRITE P*SALARY '***SALARY PLUS BONUS***'
          P*BONUS  '(IN US DOLLARS)'
    SKIP 1
  END-READ
  END
```

```
Page      1                                                97-08-19 18:14:11

          NAME                 CURRENT            INCOME
                              POSITION
  -------------------- ------------------------- ----------

  POORE JR             SECRETARY                      25000
                                                          0
                                                ***SALARY PLUS BONUS***
                                                (IN US DOLLARS)

  PREPARATA            MANAGER                        46000
                                                       9000
                                                ***SALARY PLUS BONUS***
                                                (IN US DOLLARS)

  MARKUSH              TRAINEE                        22000
                                                          0
                                                ***SALARY PLUS BONUS***
                                                (IN US DOLLARS)
```

Further Example of DISPLAY VERT with WRITE Statement:

See program WRITEX10 in library SYSEXPG.

